**详细设计文档**

|  |  |
| --- | --- |
| **课程名称：** | **企业软件项目实训** |
| **项目名称：** | **PetShop** |
| **学生姓名：** | **队长：**  **队员：** |
| **学生专业：** | **软件工程** |
| **开课学期：** | **2018-2019第二学期** |

**软件学院**

**2019年6月**

**项目详细设计文档需包含如下内容：**

需求概述

需求分析

用户角色分析

管理员

参与的业务

用户

功能性需求

新建用户

非功能性需求

安全性

隐私性

系统设计

新建用户

智能合约

界面

测试

逻辑部署方案

物理部署方案

注意：

1. 正文部分（标题行用小四号字加粗，正文内容用小四号字）
2. 图表清晰

**正式上交本设计文档时，请删除本页。**

目录

1. 项目概述

1.1 项目名称 X

1.2 建设目标 X

2. 需求分析

2.1 需求概述 X

2.1.1 目的 X

2.1.2 范围 X

2.2 用户角色分析 X

2.2.1 用户概要 X

2.2.2 涉众简档 X

2.3 功能性需求 X

2.3.1 系统功能架构 X

2.3.2 注册 X

2.3.3 登录 X

2.3.4 查看个人信息 X

2.3.5 创建宠物 X

2.3.6 修改宠物名称 X

2.3.7 修改宠物价格 X

2.3.8 上架&下架 X

2.3.9 购买宠物 X

2.3.10 查看订单 X

2.3.11 申诉 X

2.3.12 开户 X

2.3.13 仲裁 X

2.4 非功能性需求 X

3. 系统设计 X

3.1 智能合约设计 X

3.2 界面设计 X

3.3 测试设计 X

4. 逻辑部署方案 X

5. 物理部署方案 X

6. 附录 X

**1.项目概述**

**1.1项目名称：**宠物商店：PetStore

**1.2建设目标：**前端基于vue框架开发，后端采用solidity智能合约，什么什么基于以太坊区块链？目的是开发一个宠物售卖和转卖分布式市场。

**2.需求分析**

**2.1需求概述**

**2.1.1 目的**

本部分说明宠物商店的需求模块，主要阐述了出售人、购买人和管理员需要进行的各种例行操作及异常处理操作，阐明软件的功能与非功能需求，便于开发人员根据需求进行开发设计。

**2.1.2 范围**

适用于宠物商店的所有相关模块的设计及实现。

**2.2用户角色分析**

**2.2.1 用户概要**

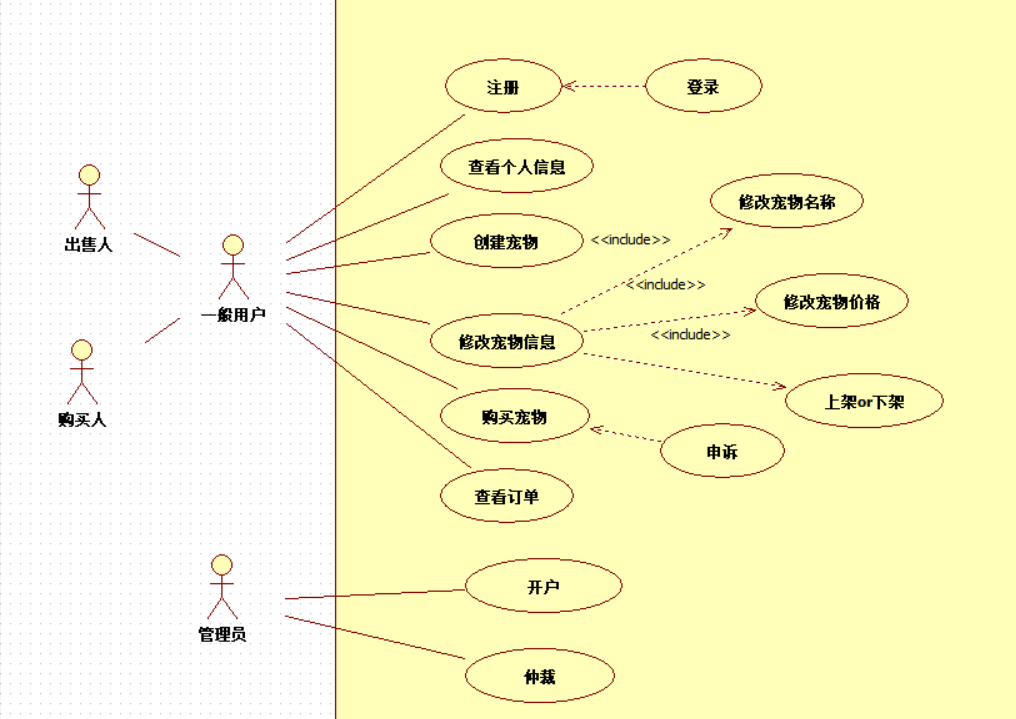
|  |  |  |
| --- | --- | --- |
| **名称** | **说明** | **涉及角色** |
| 一般用户 | 本质是自然人，其基本属性是名字和帐户余额，以及名下的宠物列表。  人可以创建一个宠物，一个宠物被创建后，挂接到一个人的名下，在被失效之前，就一直有效，宠物被创建时并没有价格 | 购买人、出售人 |
| 监督员 | 帐户开户，初始化，监测市场里的宠物价格分布，售卖次数，处理纠纷 | 管理员 |

**2.2.2 角色简档**

|  |  |
| --- | --- |
| **名称** | **说明** |
| 出售人 | 将宠物上架，制定价格，售卖成功则收款 |
| 购买人 | 选择宠物，进行购买，初始资金分配10000元 |
| 宠物 | ID，名称，品类（猫，狗，兔等），出生日期，价格，描述（一段文字供展示）等,，有效状态,图片所在url等扩展信息 |
| 管理员 | 帐户开户，初始化，监测市场里的宠物价格分布，售卖次数，处理纠纷 |
| 市场 | 展示在售宠物列表 |

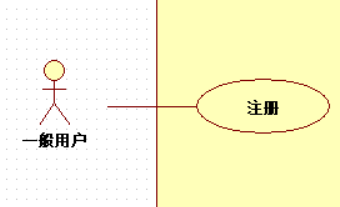
**2.3功能性需求**

**2.3.1系统功能架构**



**2.3.2 注册**

**1.用例图**



1. **用例描述**

**2.1用例简要描述**

该用例规定了自然人如何注册成为一般用户

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户处于初始界面

**2.4用例后置条件**

注册申请成功

等待管理员开户

**2.5事件流**

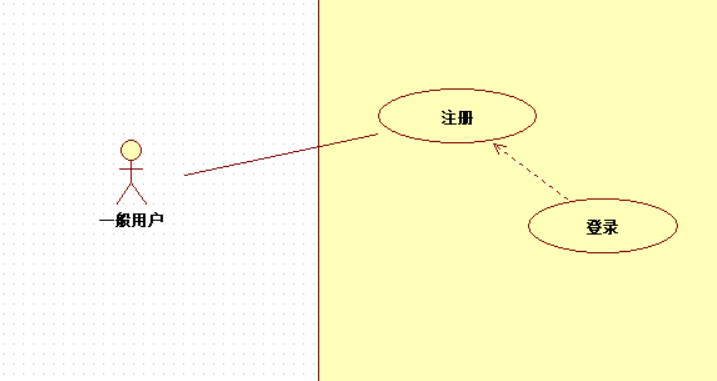
1. 用户输入姓名点击确定

**2.6 用例相关业务数据**

用户姓名、操作者地址

**2.3.3 登录**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了一般用户如何登录至宠物售卖系统

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户处于初始界面；

用户拥有一个开户的账户

**2.4用例后置条件**

登陆成功，跳转至主界面

**2.5事件流**

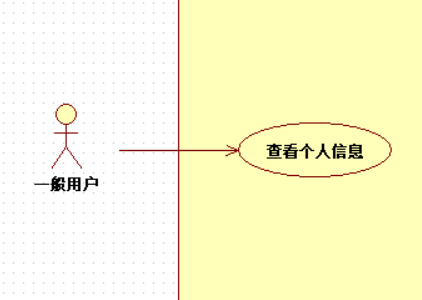
1. 用户输入账号密码点击确定
2. 跳转至主页面

**2.6用例相关业务数据**

申请信息、操作者地址

**2.3.4 查看个人信息**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了一般用户如何查看自己账户的基本信息

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户已登录；

一般用户处于主界面

**2.4用例后置条件**

一般用户处于个人信息页面

**2.5事件流**

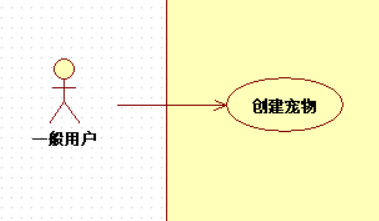
1. 点击“个人信息”索引
2. 进入到个人信息页面

**2.6用例相关业务数据**

操作者地址

**2.3.5 创建宠物**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了用户如何上传宠物

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户已开户；

**2.4用例后置条件**

一般用户处于宠物信息页面

**2.5事件流**

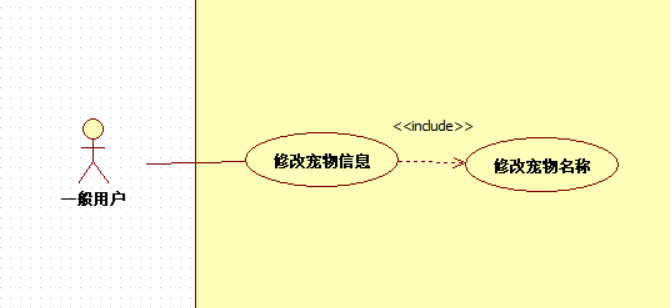
1. 点击进入到宠物页面
2. 点击创建宠物
3. 输入宠物基本信息并确定
4. 宠物信息显示在宠物页面

**2.6用例相关业务数据**

宠物姓名、宠物种类、宠物生日、宠物描述、宠物图片地址、操作者地址、宠物编号

**2.3.6 修改宠物名称**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了用户如何修改已经上传了的宠物的名称

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户已拥有创建的宠物

处于宠物信息页面

**2.4用例后置条件**

一般用户处于宠物信息页面

**2.5事件流**

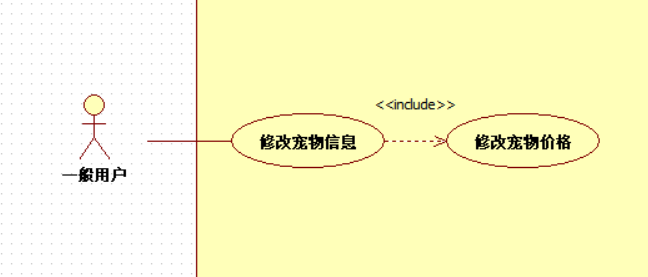
1. 点击要更改名字的宠物
2. 点击更改名字
3. 输入新名字并确定

**2.6用例相关业务数据**

宠物编号、宠物姓名、操作者地址

**2.3.7 修改宠物价格**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了用户如何修改已经上传了的宠物的价格

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户已拥有创建的宠物

用户处于宠物信息页面

**2.4用例后置条件**

一般用户处于宠物信息页面

**2.5事件流**

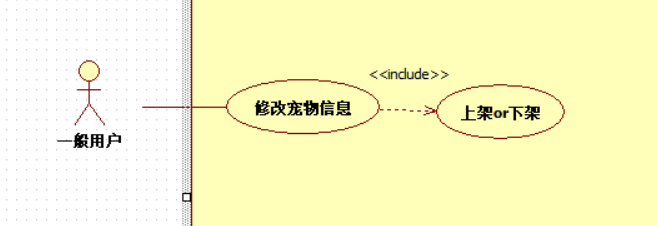
1. 点击价格
2. 输入宠物价格并点击确定

**2.6用例相关业务数据**

宠物编号、宠物价格、操作者地址

**2.3.8 上架or下架**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了用户如何将自己的宠物上架或下架

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户已拥有创建的宠物

用户处于宠物信息页面

**2.4用例后置条件**

一般用户处于宠物信息页面

**2.5事件流**

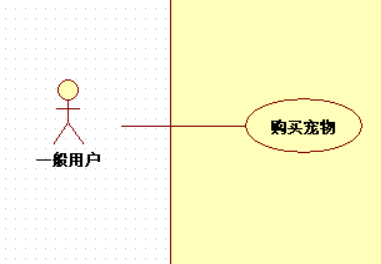
1. 点击要选择的宠物的上架/下架按键
2. 自动完成切换上下架状态操作

**2.6用例相关业务数据**

宠物编号、宠物状态、操作者地址

**2.3.9 购买宠物**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了用户如何购买他人上架的宠物

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户已开户；

市场里有他人上架的宠物；

自己的账户余额不小于要买宠物的钱价格

**2.4用例后置条件**

跳转出购买成功提示

**2.5事件流**

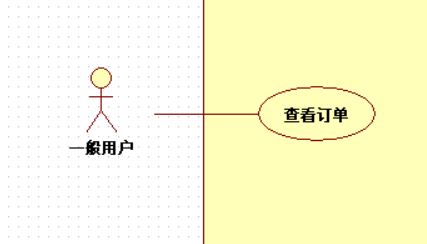
1. 进入市场
2. 点击进入要购买的宠物的详情页面
3. 点击购买
4. 若资金大于宠物价格则交易完成

**2.6用例相关业务数据**

购买者地质、售卖者地址、宠物编号、宠物价格、购买者余额、售卖者余额、交易编号、订单编号

**2.3.10 查看订单**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了用户如何查看自己的订单

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户已登录

**2.4用例后置条件**

跳转至订单页面

**2.5事件流**

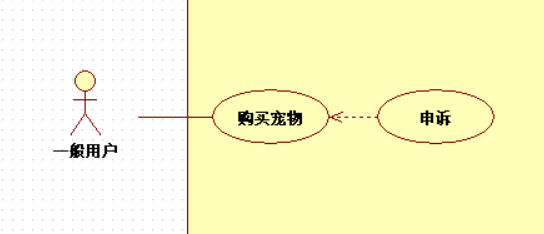
1. 点击订单索引
2. 进入到订单信息页面

**2.6用例相关业务数据**

交易编号、订单编号、购买者地质、售卖者地址、宠物编号、宠物价格

**2.3.11 申诉**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了用户如何申诉订单

**2.2用例角色**

一般用户

**2.3用例前置条件**

一般用户已达成一笔交易（拥有订单记录，无论买卖）

**2.4用例后置条件**

提示申诉成功；

等待管理员仲裁

**2.5事件流**

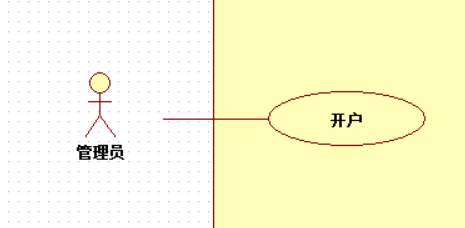
1. 进入订单信息页面
2. 点击要申诉的订单的“申诉”按键
3. 自动发送申诉请求，等待管理员审核

**2.6用例相关业务数据**

订单编号、申诉信息、购买者地质、售卖者地址、宠物编号、宠物价格

**2.3.12 开户**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了管理员如何给用户开户

**2.2用例角色**

管理员

**2.3用例前置条件**

有用户申请注册（开户）

**2.4用例后置条件**

用户开户成功

**2.5事件流**

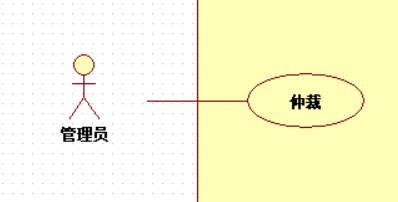
1. 点击用户申请页面
2. 确认申请信息则完成开户

**2.6用例相关业务数据**

申请者编号、申请者地址、操作者地址、管理员地址、申请者状态、申请者余额

**2.3.12 仲裁**

**1.用例图**



**2.用例描述**

**2.1用例简要描述**

该用例规定了管理员如何仲裁一笔交易

**2.2用例角色**

管理员

**2.3用例前置条件**

有用户申诉一笔交易

**2.4用例后置条件**

仲裁成功则交易回滚

或拒绝仲裁交易不回滚

**2.5事件流**

1. 点击申请页面
2. 确认申诉信息则自动完成仲裁回滚

**2.6用例相关业务数据**

交易编号、订单编号、申诉信息、购买者地质、售卖者地址、宠物编号、宠物价格、申诉状态、仲裁状态

**2.4非功能性需求**

1. **性能**

在95％的情况下，一般时段响应时间不超过3秒，高峰时段不超过6秒。

支持2000用户，支持MB级数据。

1. **安全性**

严格权限访问控制，在账户未开户状态下用户无法进入主页

交易已经完成交易记录便会存在区块链中，无法被更改无法被伪造

1. **可靠性**

对输入有提示，数据有检查，防止数据异常。

系统健壮性强，应该能处理系统运行过程中出现的各种异常情况，如：人为操作错误、输入非法数据、硬件设备失败等。

要求系统7x24小时运行，全年持续运行故障停运时间累计不能超过3天。

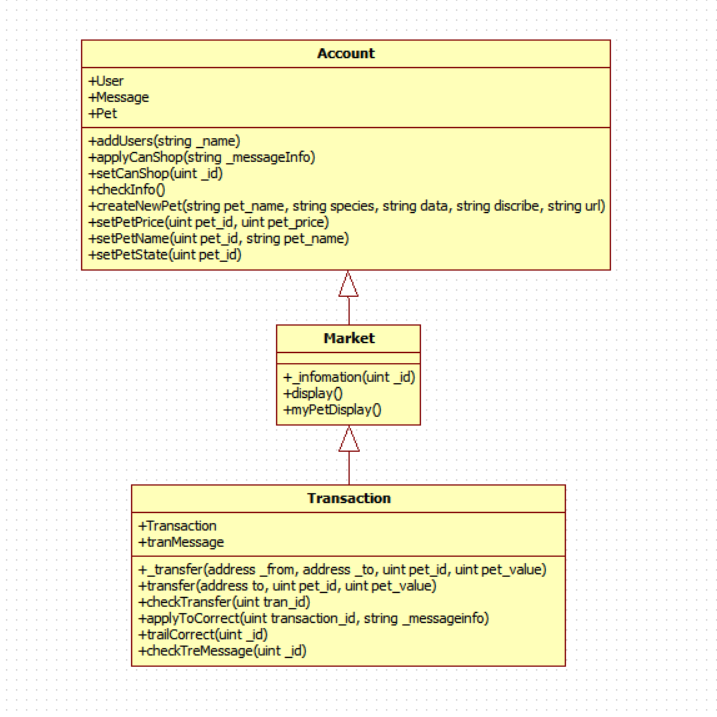
1. **易用性**

界面简洁美观

1. **系统设计**

**3.1智能合约设计**

**3.1.1合约模型**



**3.1.2 合约描述**

**Account：**

|  |  |  |  |
| --- | --- | --- | --- |
| 合约名 | Account | | |
| 合约描述 | 该合约用于用户申请账户，申请开户，管理员开户以及开户用户的部分操作，如查看个人信息，创建宠物、修改宠物信息以及上架或下架宠物。 | | |
| 内部参数 | 类型 | 描述 | |
| User | Struct | 存储用户姓名和开户状态 | |
| Message | Struct | 存储信息内容、来源、编号、状态 | |
| Pet | Struct | 存储宠物编号、姓名、种类、生日、价格、描述、图片地址、上架状态 | |
|  | | | |
| 方法名 | 参数及说明 | 返回值 | 方法描述 |
| addUsers | 用户姓名 | 无 | 创建未开户账户，生成一个账户编号 |
| applyCanShop | String信息 | 无 | 用户申请开户 |
| setCanShop | 用户id | 无 | 管理员设置用户开户，更改用户状态，并分配1000余额 |
| checkInfo | 默认传入操作者地址 | 姓名、余额、开户状态 | 查看操作者个人信息 |
| createNewPet | 宠物姓名、种类、生日、描述、图片地址 | 无 | 操作者创建宠物，默认设置上架状态为false，价格为0，分配宠物编号 |
| setPetPrice | 宠物编号及价格 | 无 | 修改操作者持有的宠物价格 |
| setPetName | 宠物编号及姓名（string） | 无 | 修改操作者持有的宠物姓名 |
| setPetState | 宠物编号 | 无 | 修改操作者持有的宠物状态 |

**Market：**

|  |  |  |  |
| --- | --- | --- | --- |
| 合约名 | Market | | |
| 合约描述 | 该合约用于检索所有上架的宠物并将必要信息返回，通过接口将宠物信息展示在前端页面中，及操作者查看自己的宠物 | | |
| 方法名 | 参数及说明 | 返回值 | 方法描述 |
| \_information | 宠物id | 宠物姓名、种类、生日、价格、描述、图片地址 | 私有函数，用于返回制定编号的宠物信息 |
| display | 无 | 无 | 循环调用私有函数\_information，返回所有上架的宠物信息 |
| myPetDisplay | 默认传入操作者地址 | 无 | 循环调用私有函数\_information，  返回所有隶属于操作者的已上架的宠物信息 |

**Transaction：**

|  |  |  |  |
| --- | --- | --- | --- |
| 合约名 | Transaction | | |
| 合约描述 | 用户通过该合约购买上架的宠物，或者对订单进行申诉  管理员通过该合约仲裁某比交易 | | |
| 内部参数 | 类型 | 描述 | |
| Transaction | Struct | 交易信息，包含：交易编号、购买者地址、售卖者地址、宠物编号、宠物价格 | |
| TranMessage | Struct | 申诉信息，包含：冲正编号、交易编号、购买者地址、售卖者地址、宠物编号、宠物价格、申诉说明、申诉状态、冲正状态 | |
|  | | | |
| 方法名 | 参数及说明 | 返回值 | 方法描述 |
| \_transfer | 购买者地址、售卖者地址、宠物编号、宠物价格 | 无 | 私有函数，用于处理一笔交易，将宠物的持有者地址更改为购买者，将购买者及售卖者双方的金钱进行交易 |
| transfer | 售卖者地址，宠物编号、宠物价格及默认传入的操作者（购买者）地址 | 无 | 再确认了传入的宠物编号上架状态为真后，调用私有函数\_transfer进行一笔交易 |
| checkTransfer | 交易编号 | 订单编号、售卖者地址、购买者地址、宠物编号、宠物价格 | 查看一笔订单的信息 |
| applyToCorrect | 订单编号  信息 | 无 | 对某比交易申请申诉，交易双方均可 |
| trailCorrect | 交易编号 | 无 | 管理员对某笔交易进行回滚，在后台交换交易双方角色，进行反交易 |
| checkTraMessage | 交易订单编号 | 冲正编号、交易编号、售卖者地址、购买者地址、宠物编号、宠物价格、申诉信息、申诉状态、冲正状态 | 检查一笔申诉是否成功冲正 |

**3.2界面设计**